

flysun的博客

RSS订阅

自己的学习总结，大家觉得不错可以拿去借鉴。

ActiveMQ的环境搭建及使用

2016年10月24日 00:08:22

阅读数：4205



0



写评论



目录



收藏



微信



微博



QQ

一：环境搭建

首先需要下载ActiveMQ的开发包：

apache-activemq-5.11.1-bin.zip	2015/6/13 15:21	WinRAR ZIP 压缩文...	47,423 KB
--------------------------------	-----------------	-------------------	-----------

解压之后，目录结构如下所示：

名称	修改日期	类型	大小
bin	2015/2/13 11:05	文件夹	
conf	2015/2/13 11:05	文件夹	
data	2016/10/23 23:18	文件夹	
docs	2015/2/13 11:05	文件夹	
examples	2015/2/13 11:05	文件夹	
lib	2015/2/13 11:05	文件夹	
webapps	2015/2/13 11:05	文件夹	
webapps-demo	2015/2/13 11:05	文件夹	
activemq-all-5.11.1.jar	2015/2/13 11:01	Executable Jar File	6,437 KB
LICENSE	2015/2/13 11:05	文件	41 KB
NOTICE	2015/2/13 11:05	文件	4 KB
README.txt	2015/2/13 11:05	文本文档	3 KB

运行bin目录下的activemq.bat命令，启动ActiveMQ服务。

然后：http://127.0.0.1:8161/admin/ 验证服务启动情况，用户名/密码 admin/admin。界面如下：

登录

注册

X

127.0.0.1:8161/admin/index.jsp

应用 百度 资料 医联预约平台 CSDN博客 work 临时 系统



Home | Queues | Topics | Subscribers | Connections | Network | Scheduled | Send

Welcome!

Welcome to the Apache ActiveMQ Console of **localhost** (ID:Lenovo-flySun-49621-1477235917191-0:1)

You can find more information about Apache ActiveMQ on the [Apache ActiveMQ Site](#)

Broker

Name	localhost
Version	5.11.1
ID	ID:Lenovo-flySun-49621-1477235917191-0:1
Uptime	37 minutes
Store percent used	0
Memory percent used	0
Temp percent used	0

Copyright 2005-2014 The Apache Software Foundation.

二：ActiveMQ使用

引入上面解压包中的activemq-all-5.11.1.jar

1. ActiveMQ 点对点消息实现

1.1 直接receive方式

消息生产者：

```
1  import javax.jms.Connection;
2  import javax.jms.ConnectionFactory;
3  import javax.jms.Destination;
4  import javax.jms.JMSException;
5  import javax.jms.MessageProducer;
6  import javax.jms.Session;
7  import javax.jms.TextMessage;
8
9  import org.apache.activemq.ActiveMQConnection;
10 import org.apache.activemq.ActiveMQConnectionFactory;
11
12 /**
13  * 消息生产者
14  * @author Administrator
15  *
16
17
18 19  private static final String USERNAME=ActiveMQConnection.DEFAULT_USER; // 默认的连接用户名
19  private static final String PASSWORD=ActiveMQConnection.DEFAULT_PASSWORD; // 默认的连接密码
20  private static final String BROKEURL=ActiveMQConnection.DEFAULT_BROKER_URL; // 默认的连接地址
21  private static final int SENDNUM=10; // 发送的消息数量
22
23
24  public static void main(String[] args) {
25
26      ConnectionFactory connectionFactory; // 连接工厂
27      Connection connection = null; // 连接
28      Session session; // 会话 接受或者发送消息的线程
29      Destination destination; // 消息的目的地
30      MessageProducer messageProducer; // 消息生产者
31
32      // 实例化连接工厂
33      connectionFactory=new ActiveMQConnectionFactory(JMSProducer.USERNAME, JMSProducer.PASSWORD, JMSProducer.BROKEURL);
34
35
36      try {
37          connection=connectionFactory.createConnection(); // 通过连接工厂获取连接
38          connection.start(); // 启动连接
39          session=session.createSession(Boolean.TRUE, Session.AUTO_ACKNOWLEDGE); // 创建Session
40          destination=session.createQueue("FirstQueue1"); // 创建消息队列
41          messageProducer=session.createProducer(destination); // 创建消息生产者
42          sendMessage(session, messageProducer); // 发送消息
43          session.commit();
44      } catch (Exception e) {
45          // TODO Auto-generated catch block
46          e.printStackTrace();
47      } finally{
48          if(connection!=null){
49              try {
50                  connection.close();
51              } catch (JMSException e) {
52                  // TODO Auto-generated catch block
53                  e.printStackTrace();
54              }
55          }
56      }
57
58  /**
```

登录 注册 ×

```

59     * 发送消息
60     * @param session
61     * @param messageProducer
62     * @throws Exception
63     */
64     public static void sendMessage(Session session, MessageProducer messageProducer) throws Exception {
65         for (int i = 0; i < JMSProducer.SENDNUM; i++) {
66             TextMessage message = session.createTextMessage("ActiveMQ 发送的消息" + i);
67             System.out.println("发送消息: " + "ActiveMQ 发送的消息" + i);
68             messageProducer.send(message);
69         }
70     }
71 }

```

消息消费者:

```

1  import javax.jms.Connection;
2  import javax.jms.ConnectionFactory;
3  import javax.jms.Destination;
4  import javax.jms.JMSException;
5  import javax.jms.MessageConsumer;
6  import javax.jms.Session;
7  import javax.jms.TextMessage;
8
9  import org.apache.activemq.ActiveMQConnection;
10 import org.apache.activemq.ActiveMQConnectionFactory;
11

```

[登录](#)

[注册](#)

×

```

14  * @author Administrator15 | *
15  */
16
17 public class JMSConsumer {
18
19     private static final String USERNAME = ActiveMQConnection.DEFAULT_USER; // 默认的连接用户名
20     private static final String PASSWORD = ActiveMQConnection.DEFAULT_PASSWORD; // 默认的连接密码
21     private static final String BROKEURL = ActiveMQConnection.DEFAULT_BROKER_URL; // 默认的连接地址
22
23     public static void main(String[] args) {
24         ConnectionFactory connectionFactory; // 连接工厂
25         Connection connection = null; // 连接
26         Session session; // 会话 接受或者发送消息的线程
27         Destination destination; // 消息的目的地
28         MessageConsumer messageConsumer; // 消息的消费者
29
30         // 实例化连接工厂
31         connectionFactory = new ActiveMQConnectionFactory(JMSConsumer.USERNAME, JMSConsumer.PASSWORD, JMSConsumer.BROKEURL);
32
33         try {
34             connection = connectionFactory.createConnection(); // 通过连接工厂获取连接
35             connection.start(); // 启动连接
36             session = connection.createSession(Boolean.FALSE, Session.AUTO_ACKNOWLEDGE); // 创建Session
37             destination = session.createQueue("FirstQueue1"); // 创建连接的消息队列
38             messageConsumer = session.createConsumer(destination); // 创建消息消费者
39             while (true) {
40                 TextMessage textMessage = (TextMessage) messageConsumer.receive(100000);
41                 if (textMessage != null) {
42                     System.out.println("收到的消息: " + textMessage.getText());
43                 } else {
44                     break;
45                 }
46             }
47         } catch (JMSException e) {
48             // TODO Auto-generated catch block
49             e.printStackTrace();
50         }
51     }
52 }

```

1.2 使用Listener监听方式

生产者代码同上.

监听器类:

```

1  import javax.jms.JMSException;
2  import javax.jms.Message;
3  import javax.jms.MessageListener;
4  import javax.jms.TextMessage;
5
6  /**
7   * 消息监听
8   * @author Administrator
9   *
10  */
11 public class Listener implements MessageListener {

```



0



写评论



目录



收藏



微信



微博



QQ

```
12
13     @Override
14     public void onMessage(Message message) {
15         // TODO Auto-generated method stub
16         try {
17             System.out.println("收到的消息: "+((TextMessage)message).getText());
18         } catch (JMSException e) {
19             // TODO Auto-generated catch block
```

登录

注册

×

```
23
24 | }
```

消息消费者中注册监听器:

```
1 import javax.jms.Connection;
2 import javax.jms.ConnectionFactory;
3 import javax.jms.Destination;
4 import javax.jms.JMSException;
5 import javax.jms.MessageConsumer;
6 import javax.jms.Session;
7 import javax.jms.TextMessage;
8
9 import org.apache.activemq.ActiveMQConnection;
10 import org.apache.activemq.ActiveMQConnectionFactory;
11
12 /**
13  * 消息消费者
14  * @author Administrator
15  *
16  */
17 public class JMSConsumer2 {
18
19     private static final String USERNAME=ActiveMQConnection.DEFAULT_USER; // 默认的连接用户名
20     private static final String PASSWORD=ActiveMQConnection.DEFAULT_PASSWORD; // 默认的连接密码
21     private static final String BROKEURL=ActiveMQConnection.DEFAULT_BROKER_URL; // 默认的连接地址
22
23     public static void main(String[] args) {
24         ConnectionFactory connectionFactory; // 连接工厂
25         Connection connection = null; // 连接
26         Session session; // 会话 接受或者发送消息的线程
27         Destination destination; // 消息的目的地
28         MessageConsumer messageConsumer; // 消息的消费者
29
30         // 实例化连接工厂
31         connectionFactory=new ActiveMQConnectionFactory(JMSConsumer2.USERNAME, JMSConsumer2.PASSWORD, JMSConsumer2.BROKEURL);
32
33         try {
34             connection=connectionFactory.createConnection(); // 通过连接工厂获取连接
35             connection.start(); // 启动连接
36             session=connection.createSession(Boolean.FALSE, Session.AUTO_ACKNOWLEDGE); // 创建Session
37             destination=session.createQueue("FirstQueue1"); // 创建连接的消息队列
38             messageConsumer=session.createConsumer(destination); // 创建消息消费者
39             messageConsumer.setMessageListener(new Listener()); // 注册消息监听
40         } catch (JMSException e) {
41             // TODO Auto-generated catch block
42             e.printStackTrace();
43         }
44     }
45 }
```

登录

注册

×

```
12 /**
13  * 消息生产者-消息发布者
14  * @author Administrator
15  *
16  */
17 public class JMSProducer {
18
19     private static final String USERNAME=ActiveMQConnection.DEFAULT_USER; // 默认的连接用户名
20     private static final String PASSWORD=ActiveMQConnection.DEFAULT_PASSWORD; // 默认的连接密码
```



```
21 private static final String BROKEURL=ActiveMQConnection.DEFAULT_BROKER_URL; // 默认的连接地址
22 private static final int SENDNUM=10; // 发送的消息数量
23
24 public static void main(String[] args) {
25
26     ConnectionFactory connectionFactory; // 连接工厂
27     Connection connection = null; // 连接
28     Session session; // 会话 接受或者发送消息的线程
29     Destination destination; // 消息的目的地
30     MessageProducer messageProducer; // 消息生产者
31
32     // 实例化连接工厂
33     connectionFactory=new ActiveMQConnectionFactory(JMSProducer.USERNAME, JMSProducer.PASSWORD, JMSProducer.BROKEURL);
34
35     try {
36         connection=connectionFactory.createConnection(); // 通过连接工厂获取连接
37         connection.start(); // 启动连接
38         session=connection.createSession(Boolean.TRUE, Session.AUTO_ACKNOWLEDGE); // 创建Session
39         // destination=session.createQueue("FirstQueue1"); // 创建消息队列
40         <span style="background-color: rgb(255, 255, 102);">destination=session.createTopic("FirstTopic1");</span>
41         messageProducer=session.createProducer(destination); // 创建消息生产者
42         sendMessage(session, messageProducer); // 发送消息
43         session.commit();
44     } catch (Exception e) {
45         // TODO Auto-generated catch block
46         e.printStackTrace();
47     } finally{
48         if(connection!=null){
49             try {
50                 connection.close();
51             } catch (JMSException e) {
52                 // TODO Auto-generated catch block
53                 e.printStackTrace();
54             }
55         }
56     }
57 }
58
59 /**
60  * 发送消息
61  * @param session
62  * @param messageProducer
63  * @throws Exception
64  */
65 public static void sendMessage(Session session,MessageProducer messageProducer)throws Exception{
66     for(int i=0;i<JMSProducer.SENDNUM;i++){
67         TextMessage message=session.createTextMessage("ActiveMQ 发送的消息"+i);
68         System.out.println("发送消息: "+"ActiveMQ 发布的消息"+i);
69         messageProducer.send(message);
70     }
71 }
72 }
```

消息的监听器一：

```
1 import javax.jms.JMSException;
2 import javax.jms.Message;
```

登录 注册 ✕

```
6 /**
7  * 消息监听-订阅者一
8  * @author Administrator
9  *
10 */
11 public class Listener implements MessageListener{
12
13     @Override
14     public void onMessage(Message message) {
15         // TODO Auto-generated method stub
16         try {
17             System.out.println("订阅者一收到的消息: "+((TextMessage)message).getText());
18         } catch (JMSException e) {
19             // TODO Auto-generated catch block
20             e.printStackTrace();
21         }
22     }
23 }
24 }
```



消息的监听器二：

```
1 import javax.jms.JMSException;
2 import javax.jms.Message;
3 import javax.jms.MessageListener;
4 import javax.jms.TextMessage;
5
```



```
6  /**
7   * 消息监听-订阅者二
8   * @author Administrator
9   *
10  */
11  public class Listener2 implements MessageListener{
12
13      @Override
14      public void onMessage(Message message) {
15          // TODO Auto-generated method stub
16          try {
17              System.out.println("订阅者二收到的消息: "+((TextMessage)message).getText());
18          } catch (JMSException e) {
19              // TODO Auto-generated catch block
20              e.printStackTrace();
21          }
22      }
23
24  }
```

消息的订阅者一:

```
1  import javax.jms.Connection;
2  import javax.jms.ConnectionFactory;
3  import javax.jms.Destination;
4  import javax.jms.JMSException;
5  import javax.jms.MessageConsumer;
6  import javax.jms.Session;
7
8  import org.apache.activemq.ActiveMQConnection;
9  import org.apache.activemq.ActiveMQConnectionFactory;
10
11  /**
12   * 消息消费者-消息订阅者一
13   * @author Administrator
14   *
15   */
16  public class JMSConsumer {
```

登录

注册



```
19      private static final String PASSWORD=ActiveMQConnection.DEFAULT_PASSWORD; // 默认的连接密码20
20      private static final String BROKEURL=ActiveMQConnection.DEFAULT_BROKER_URL; // 默认的连接地址21
21      public static void main(String[] args) {
22          ConnectionFactory connectionFactory; // 连接工厂
23          Connection connection = null; // 连接
24          Session session; // 会话 接受或者发送消息的线程
25          Destination destination; // 消息的目的地
26          MessageConsumer messageConsumer; // 消息的消费者
27
28          // 实例化连接工厂
29          connectionFactory=new ActiveMQConnectionFactory(JMSConsumer.USERNAME, JMSConsumer.PASSWORD, JMSConsumer.BROKEURL);
30
31          try {
32              connection=connectionFactory.createConnection(); // 通过连接工厂获取连接
33              connection.start(); // 启动连接
34              session=session.createSession(Boolean.FALSE, Session.AUTO_ACKNOWLEDGE); // 创建Session
35              // destination=session.createQueue("FirstQueue1"); // 创建连接的消息队列
36              <span style="background-color: rgb(255, 255, 102);">destination=session.createTopic("FirstTopic1");</span>
37              messageConsumer=session.createConsumer(destination); // 创建消息消费者
38              <span style="background-color: rgb(255, 255, 102);">messageConsumer.setMessageListener(new Listener()); // 注册消息监听</span>
39          } catch (JMSException e) {
40              // TODO Auto-generated catch block
41              e.printStackTrace();
42          }
43      }
44  }
```



0



写评论



目录



收藏



微信



微博



消息的订阅者二:

```
1  import javax.jms.Connection;
2  import javax.jms.ConnectionFactory;
3  import javax.jms.Destination;
4  import javax.jms.JMSException;
5  import javax.jms.MessageConsumer;
6  import javax.jms.Session;
7
8  import org.apache.activemq.ActiveMQConnection;
9  import org.apache.activemq.ActiveMQConnectionFactory;
10
11  /**
12   * 消息消费者-消息订阅者二
13   * @author Administrator
14   *
15   */
16  public class JMSConsumer2 {
17
18      private static final String USERNAME=ActiveMQConnection.DEFAULT_USER; // 默认的连接用户名
```

```
19         private static final String PASSWORD=ActiveMQConnection.DEFAULT_PASSWORD; // 默认的连接密码
20         private static final String BROKEURL=ActiveMQConnection.DEFAULT_BROKER_URL; // 默认的连接地址
21
22         public static void main(String[] args) {
23             ConnectionFactory connectionFactory; // 连接工厂
24             Connection connection = null; // 连接
25             Session session; // 会话 接受或者发送消息的线程
26             Destination destination; // 消息的目的地
27             MessageConsumer messageConsumer; // 消息的消费者
28
29             // 实例化连接工厂
30             connectionFactory=new ActiveMQConnectionFactory(JMSConsumer2.USERNAME, JMSConsumer2.PASSWORD, JMSConsumer2.BROKEURL);
31
32             try {
33                 connection=connectionFactory.createConnection(); // 通过连接工厂获取连接
34                 connection.start(); // 启动连接
35                 session=connection.createSession(Boolean.FALSE, Session.AUTO_ACKNOWLEDGE); // 创建Session
36                 // destination=session.createQueue("FirstQueue1"); // 创建连接的消息队列
37                 <span style="background-color: rgb(255, 255, 102);">destination=session.createTopic("FirstTopic1");</span>
38
39                 // 注册消息监听
40                 session.createConsumer(destination).setMessageListener(new JMSConsumer2());
41                 // TODO Auto-generated catch block
42                 e.printStackTrace();
43             }
44         }
45     }
```

登录

注册

×

^2()); // 注册消息监听



0

版权声明：本文为博主原创文章，未经博主允许不得转载。 <https://blog.csdn.net/llsun3344/article/details/52906132>

个人分类： java消息



写评论

上一篇 枚举Enum的使用



目录

下一篇 System的getenv 和GetProperty



收藏



微信



微博



QQ

想对作者说点什么?

我来说一句

ActiveMQ环境搭建



100

环境搭建

activeMQ搭建和简单使用



649

最近闲暇的时候在看一本书，关于分布式系统的，作者从使用者的角度和开发者的角度讲述了分布式系统的...

activeMQ搭建和简单使用 - CSDN博客

2018-7-19

峰策略时,提到了一种通过异步调用来实现消峰的方式——使用消息队列MQ(message ...一:环境搭建首先需要下...

ActiveMQ使用步骤--- 开发测试环境搭建及Java用例 - CSDN博客

2018-7-19

ActiveMQ的环境搭建及使用 一:环境搭建首先需要下载ActiveMQ的开发包: 解压之后,目录结构如下所示: 运行bin...

基于maven的Spring+ActiveMQ整合Demo



8618

本文主要是示范基于Maven的ActiveMQ+Spring的简单使用，基于ActiveMQ消息代理的Spring JMS消息配置...

ActiveMQ环境搭建及实例详解 - 基督山伯爵的博客 - CSDN博客

2018-1-28

ActiveMQ环境搭建及实例详解 前言: JMS(Java Messaging Service)是Java平台上有关面向消息中间件的技术规...

ActiveMQ环境搭建 - CSDN博客

2018-3-11

就像JDBC抽象了关系数据库访问、具体的实现由不同的消息中间件厂商提供,比如Apache ActiveMQ就是JMS规...

ActiveMQ使用步骤--- 开发测试环境搭建及Java用例



1577

Linux 系统 ActiveMQ 版本：5.8.0 一、下载ActiveMQ5.8.0 ActiveMQ版本地址： <http://activemq.apache.org/>...

ActiveMQ安装部署(Windows)



6.1万

JMS（Java Messaging Service）是Java平台上有关面向消息中间件的技术规范，它便于消息系统中的Java应...